(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2002－33670
（Ｐ２００２－３３６７０Ａ）
(43)公開日　平成14年１月31日(2002.1.31)

(51)Int.Cl.[7]	識別記号	FI			テーマコード（参考）
H03M 13/39		H03M 13/39			5B001
G06F 11/10	320	G06F 11/10	320	K	5D045
	330		330	N	5J065
			330	S	
G10L 19/00		G10L 9/18		A	

審査請求　未請求　請求項の数70　ＯＬ　（全14頁）

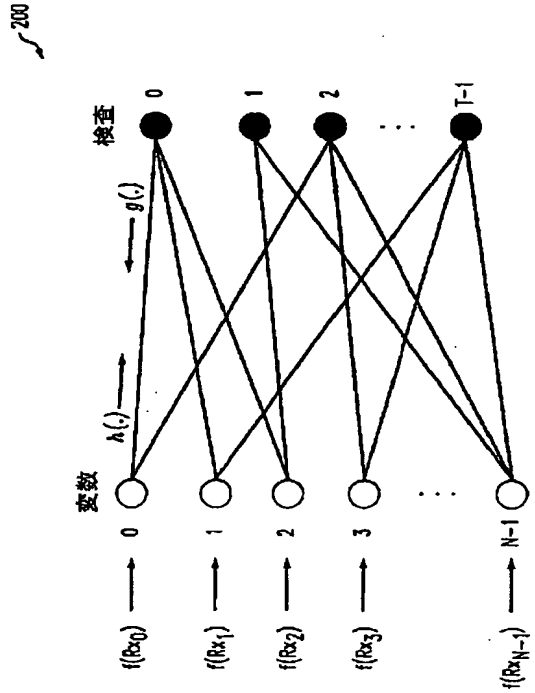
(21)出願番号　特願2001－156375(Ｐ2001－156375)
(22)出願日　平成13年５月25日(2001.5.25)
(31)優先権主張番号　０９／５７９２１６
(32)優先日　平成12年５月26日(2000.5.26)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　301030605
アギア　システムズ　ガーディアン　コーポレーション
Ａｇｅｒｅ　Ｓｙｓｔｅｍｓ　Ｇｕａｒｄｉａｎ　Ｃｏｒｐｏｒａｔｉｏｎ
アメリカ合衆国.32819－8698　フロリダ，オーランド，サウス　ジョン　ヤング　パークウェイ　9333
(74)代理人　100064447
弁理士　岡部　正夫　（外１１名）

最終頁に続く

(54)【発明の名称】確率依存グラフにおいて汎用の符号を復号化するための方法および装置

(57)【要約】
【課題】　種々の符号の繰返し復号化を実行するためのブロック並列復号化アルゴリズムおよび対応する復号化器アーキテクチャを提供すること。
【解決手段】　ブロック並列復号化アルゴリズムおよび対応する復号化器アーキテクチャは、確率依存グラフの形で構成される１組の相互接続される処理ノードを用いる。確率依存グラフは、ビットあるいはシンボルからなるブロックを符号化するために用いられる符号によって少なくとも部分的に特徴づけられ、その処理ノードは、復号化されることになる、ビットあるいはシンボルからなるブロックのためのブロック並列復号化プロセスを実施する。確率依存グラフには、たとえば、Ｎ個一組の変数ノードとＴ個一組の検査ノードとを含み、Ｎ個の変数ノードのうちの１つが、復号化される所与のブロックのＮ個の各ビットあるいはシンボルに関連する、２ノード型の確率依存グラフが用いられる場合がある。



1

【特許請求の範囲】

【請求項１】　受信されたビットあるいはシンボルを復号化するための装置であって、

前記ビットあるいはシンボルを符号化するために用いられる符号によって少なくとも部分的に特徴づけられる確率依存グラフの形で実装される１組の相互接続される処理ノードを備え、該処理ノードは、復号化されることになる前記ビットあるいはシンボルからなる所与のブロックのためのブロック並列復号化プロセスを実施する装置。

【請求項２】　前記確率依存グラフは、２ノード型の確率依存グラフを含む請求項１に記載の装置。

【請求項３】　前記所与のブロックは、復号化されることになるＮ個のビットあるいはシンボルを含み、前記１組の相互接続される処理ノードはさらに、Ｎ個一組の変数ノードと、Ｔ個一組の検査ノードとを含み、前記Ｎ個の変数ノードのうちの１つは、復号化されることになる前記Ｎ個のビットあるいはシンボルそれぞれと関連する請求項２に記載の装置。

【請求項４】　前記ブロック並列復号化プロセスは、Rx_i（ｉ＝０，１，．．．Ｎ－１）を前記受信されたビットあるいはシンボルとし、ｆ（.）を前記符号によって少なくとも部分的に決定される関数とするとき、関数ｆ（Rx_i）を前記変数ノードに接続される前記各検査ノードに送信することによって開始される請求項３に記載の装置。

【請求項５】　前記関数ｆ（.）は、閾値関数、線形換算関数および近似指数関数のうちの少なくとも１つを含む請求項４に記載の装置。

【請求項６】　前記ブロック並列復号化プロセスは、前記検査ノードそれぞれにおいて、該検査ノードに接続される前記変数ノードからの入力の関数ｇ（.）を計算するステップと、前記検査ノードの計算結果を前記接続される変数ノードに送信するステップとを含み、前記計算は並行して実行される請求項３に記載の装置。

【請求項７】　前記関数ｇ（.）は、パリティ検査関数、対数領域の双曲正接関数および対数領域の最大関数のうちの少なくとも１つを含む請求項６に記載の装置。

【請求項８】　前記ブロック並列復号化プロセスはさらに、前記変数ノードそれぞれにおいて、該変数ノードに接続される前記検査ノードからの入力の関数ｈ（.）を計算するステップと、前記変数ノードの計算結果を前記接続される検査ノードに送信するステップとを含み、前記変数ノード計算は並行して実行される請求項６に記載の装置。

【請求項９】　前記関数ｈ（.）は、多数決関数および平均化関数のうちの少なくとも１つを含む請求項８に記載の装置。

【請求項１０】　前記ブロック並列復号化プロセスの１回の繰返しは、前記所与のブロック内の全ての前記ビットあるいはシンボルのための更新された推定値を生成する請求項２に記載の装置。

2

【請求項１１】　前記ブロック並列復号化プロセスの１回の繰返しは、前記所与のブロック内の全ての前記ビットあるいはシンボルのための事後確率あるいは信頼性に対する更新された推定値を生成する請求項２に記載の装置。

【請求項１２】　前記ブロック並列復号化プロセスは、全ての前記検査ノードが前記符号の制約条件を満たすときに終了する請求項３に記載の装置。

【請求項１３】　前記ブロック並列復号化プロセスは、最大数だけ繰り返した後に終了する請求項２に記載の装置。

【請求項１４】　前記変数ノードにおける前記ブロック並列復号化プロセスの結果は、前記復号化されたビットあるいはシンボルの関連する事後確率あるいは信頼性の推定値を与える請求項３に記載の装置。

【請求項１５】　前記確率依存グラフは、有向確率依存グラフを含む請求項１に記載の装置。

【請求項１６】　前記１組の相互接続された処理ノードはさらに、初期レベル、少なくとも１つの中間レベルおよび最終レベルを有する複数レベルのノードを含む請求項１５に記載の装置。

【請求項１７】　前記初期レベルの前記ノードは、前記所与のブロック内の前記各ビットあるいはシンボルのための入力ノードを含む請求項１５に記載の装置。

【請求項１８】　前記レベルの前記ノードは、前記少なくとも１つの中間レベル内の前記ノードが、以前のレベルのノードからのみ入力を受信し、次に続くレベルのノードにのみ出力を供給するように構成され、順次的なブロック内依存性を有することなく前記各レベルにおいて計算を実行できるようにする請求項１５に記載の装置。

【請求項１９】　前記最終レベルの前記ノードの出力は、前記所与のブロックのための伝送されたビットあるいはシンボルの推定値を与える請求項１５に記載の装置。

【請求項２０】　前記最終レベルの前記ノードの出力は、前記所与のブロックのための伝送されたビットあるいはシンボルの推定値と、その関連する事後確率あるいは信頼性の推定値とを与える請求項１５に記載の装置。

【請求項２１】　前記ブロック並列復号化プロセスは、Rx_i（ｉ＝０、１、．．．Ｎ－１）を前記受信されたビットあるいはシンボルとし、ｆ（.）を前記符号によって少なくとも部分的に決定される関数とするとき、関数ｆ（Rx_i）を、前記初期レベル内の前記各ノードへの入力として供給することにより初期化される請求項１５に記載の装置。

【請求項２２】　前記関数ｆ（.）は、閾値関数、線形換算関数および近似指数関数のうちの少なくとも１つを含む請求項２１に記載の装置。

【請求項２３】　前記各中間レベルの前記ノードは、前記以前のレベルから到来する値の関数ｇ（.）を計算し、前記計算の結果を次のレベルのノードに渡す請求項１５に記載の装置。

【請求項２４】　前記関数ｇ（.）は、パリティ検査関数、対数領域の双曲正接関数、および対数領域の最大関数のうちの少なくとも１つを含む請求項２３に記載の装置。

【請求項２５】　前記レベルはそれぞれ、前記所与のブロック内の全ての前記ビットあるいはシンボルのための更新された推定値を生成する請求項１５に記載の装置。

【請求項２６】　前記ノードは複数のプログラム可能な計算ユニットを用いて実装され、前記プログラム可能な計算ユニットは、種々の符号のための種々のブロック並列復号化プロセスを実施するように構成を変更可能である請求項１に記載の装置。

【請求項２７】　前記符号はたたみ込み符号を含む請求項１に記載の装置。

【請求項２８】　前記符号はたたみ込み符号に少なくとも部分的に基づく符号を含む請求項１に記載の装置。

【請求項２９】　前記符号はブロック符号を含む請求項１に記載の装置。

【請求項３０】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、指示されたパリティ制約条件が満たされるか否かを判定することにより行われる請求項２に記載の装置。

【請求項３１】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、復号化された前記ワードと前記符号のパリティ検査行列との積がゼロベクトルになるか否かを判定することにより行われる請求項２に記載の装置。

【請求項３２】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、有効なコードワードへの収束を示すために以前のレベルからの入力の関数を用いて行われる請求項１５に記載の装置。

【請求項３３】　受信されたビットあるいはシンボルを復号化するための方法であって、該方法は、
前記ビットあるいはシンボルを符号化するために用いられる符号によって少なくとも部分的に特徴づけられる確率依存グラフの形で構成される１組の相互接続される処理ノードに、前記受信されたビットあるいはシンボルを適用するステップと、
前記処理ノード内で、復号化されることになる前記ビットあるいはシンボルからなる所与のブロックのためのブロック並列復号化プロセスを実施するステップとを備える方法。

【請求項３４】　前記確率依存グラフは、２ノード型の確率依存グラフを含む請求項３３に記載の方法。

【請求項３５】　前記所与のブロックは、復号化されることになるＮ個のビットあるいはシンボルを含み、前記１組の相互接続される処理ノードはさらに、Ｎ個一組の変数ノードとＴ個一組の検査ノードとを含み、前記Ｎ個の変数ノードのうちの１つは、復号化されることになる前記Ｎ個のビットあるいはシンボルそれぞれに関連する請求項３４に記載の方法。

【請求項３６】　前記ブロック並列復号化プロセスは、Rx_i（ｉ＝０、１、．．．Ｎ－１）を前記受信されたビットあるいはシンボルとし、ｆ（.）を前記符号によって少なくとも部分的に決定される関数とするとき、関数ｆ（Rx_i）を前記変数ノードに接続される前記各検査ノードに送信することにより開始される請求項３５に記載の方法。

【請求項３７】　前記関数ｆ（.）は、閾値関数、線形換算関数および近似指数関数のうちの少なくとも１つを含む請求項３６に記載の方法。

【請求項３８】　前記ブロック並列復号化プロセスは、前記検査ノードそれぞれにおいて、該検査ノードに接続される前記変数ノードからの入力の関数ｇ（.）を計算するステップと、前記検査ノードの計算結果を前記接続される変数ノードに送信するステップとを含み、前記計算は並行して実行される請求項３５に記載の方法。

【請求項３９】　前記関数ｇ（.）は、パリティ検査関数、対数領域の双曲正接関数、および対数領域の最大関数のうちの少なくとも１つを含む請求項３８に記載の方法。

【請求項４０】　前記ブロック並列復号化プロセスはさらに、前記変数ノードそれぞれにおいて、該変数ノードに接続される前記検査ノードからの入力の関数ｈ（.）を計算するステップと、前記変数ノードの計算結果を前記接続される検査ノードに送信するステップとを含み、前記変数ノード計算は並行して実行される請求項３８に記載の方法。

【請求項４１】　前記関数ｈ（.）は、多数決関数および平均化関数のうちの少なくとも１つを含む請求項４０に記載の方法。

【請求項４２】　前記ブロック並列復号化プロセスの１回の繰返しは、前記所与のブロック内の全ての前記ビットあるいはシンボルのための更新された推定値を生成する請求項３４に記載の方法。

【請求項４３】　前記ブロック並列復号化プロセスの１回の繰返しは、前記所与のブロック内の全ての前記ビットあるいはシンボルのための事後確率あるいは信頼性に対する更新された推定値を生成する請求項３４に記載の方法。

【請求項４４】　前記ブロック並列復号化プロセスは、全ての前記検査ノードが前記符号の制約条件を満たすときに終了する請求項３５に記載の方法。

【請求項４５】　前記ブロック並列復号化プロセスは、最大数だけ繰り返した後に終了する請求項３４に記載の方法。

【請求項４６】　前記変数ノードにおける前記ブロック並列復号化プロセスの結果は、前記復号化されたビットあるいはシンボルの関連する事後確率あるいは信頼性の推定値を与える請求項３５に記載の方法。

【請求項４７】　前記確率依存グラフは、有向確率依存グラフを含む請求項３３に記載の方法。

【請求項４８】　前記１組の相互接続された処理ノードはさらに、初期レベル、少なくとも１つの中間レベルおよび最終レベルを有する複数レベルのノードを含む請求項４７に記載の方法。

【請求項４９】　前記初期レベルの前記ノードは、前記所与のブロック内の前記各ビットあるいはシンボルのための入力ノードを含む請求項４７に記載の方法。

【請求項５０】　前記レベルの前記ノードは、前記少なくとも１つの中間レベル内の前記ノードが、以前のレベルのノードからのみ入力を受信し、次に続くレベルのノードにのみ出力を供給するように構成され、順次的なブロック内依存性を有することなく前記各レベルにおいて計算を実行できるようにする請求項４７に記載の方法。

【請求項５１】　前記最終レベルの前記ノードの出力は、前記所与のブロックのための伝送されたビットあるいはシンボルの推定値を与える請求項４７に記載の方法。

【請求項５２】　前記最終レベルの前記ノードの出力は、前記所与のブロックのための伝送されたビットあるいはシンボルの推定値と、その関連する事後確率あるいは信頼性の推定値とを与える請求項４７に記載の方法。

【請求項５３】　前記ブロック並列復号化プロセスは、Rx_i（ｉ＝０、１、．．．Ｎ－１）を前記受信されたビットあるいはシンボルとし、ｆ（.）を前記符号によって少なくとも部分的に決定される関数とするとき、関数ｆ（Rx_i）を、前記初期レベル内の前記各ノードへの入力として供給することにより初期化される請求項４７に記載の方法。

【請求項５４】　前記関数ｆ（.）は、閾値関数、線形換算関数および近似指数関数のうちの少なくとも１つを含む請求項５３に記載の方法。

【請求項５５】　前記各中間レベルの前記ノードは、前記以前のレベルから到来する値の関数ｇ（.）を計算し、前記計算の結果を次のレベルのノードに渡す請求項４７に記載の方法。

【請求項５６】　前記関数ｇ（.）は、パリティ検査関数、対数領域の双曲正接関数、および対数領域の最大関数のうちの少なくとも１つを含む請求項５５に記載の方法。

【請求項５７】　前記レベルはそれぞれ、前記所与のブロック内の全ての前記ビットあるいはシンボルのための更新された推定値を生成する請求項４７に記載の方法。

【請求項５８】　前記ノードは複数のプログラム可能な計算ユニットを用いて実装され、該プログラム可能な計算ユニットは、種々の符号のための種々のブロック並列復号化プロセスを実施するように構成を変更可能である請求項３３に記載の方法。

【請求項５９】　前記符号はたたみ込み符号を含む請求項３３に記載の方法。

【請求項６０】　前記符号はたたみ込み符号に少なくとも部分的に基づく符号を含む請求項３３に記載の方法。

【請求項６１】　前記符号はブロック符号を含む請求項３３に記載の方法。

【請求項６２】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、指示されたパリティ制約条件が満たされるか否かを判定することにより行われる請求項３４に記載の方法。

【請求項６３】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、復号化された前記ワードと前記符号のパリティ検査行列との積がゼロベクトルになるか否かを判定することにより行われる請求項３４に記載の方法。

【請求項６４】　復号化されたビットあるいはシンボルからなるワードが前記符号の有効なコードワードであるか否かに関する判定は、有効なコードワードへの収束を示すために以前のレベルからの入力の関数を用いて行われる請求項４７に記載の方法。

【請求項６５】　受信されたビットあるいはシンボルを復号化するための装置であって、
確率依存グラフの形に構成される１組のノードを実装するための１つあるいは複数の計算ユニットを備え、該計算ユニットは、前記ビットあるいはシンボルを符号化するために用いられる符号によって少なくとも部分的に特徴づけられ、前記１組のノードは、復号化されることになる前記ビットあるいはシンボルからなる所与のブロックにブロック並列復号化プロセスを提供するように構成される装置。

【請求項６６】　前記計算ユニットはそれぞれ、前記ノードのうちの特定の１つのノードに対応する請求項６５に記載の装置。

【請求項６７】　前記計算ユニットはそれぞれ、前記ノードのうちの多数のノードに対応する請求項６５に記載の装置。

【請求項６８】　受信されたビットあるいはシンボルを復号化するための方法であって、該方法は、
確率依存グラフの形で構成される１組のノードを実装し、かつ前記ビットあるいはシンボルを符号化するために用いられる符号によって少なくとも部分的に特徴づけられる１組の１つあるいは複数の計算ユニットに、前記

受信されたビットあるいはシンボルを適用するステップと、
前記ノード内に、復号化されることになる前記ビットあるいはシンボルからなる所与のブロックのためのブロック並列復号化プロセスを実装するステップとを備える方法。
【請求項６９】　前記計算ユニットはそれぞれ、前記ノードのうちの特定の１つのノードに対応する請求項６８に記載の方法。
【請求項７０】　前記計算ユニットはそれぞれ、前記ノードのうちの多数のノードに対応する請求項６８に記載の方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は全般に情報の符号化および復号化技術に関し、より詳細には、符号化された情報信号を復号化する際に用いるための復号化器アルゴリズムおよびアーキテクチャに関する。
【０００２】
【従来の技術】符号化は、デジタル情報を伝送する際に、ビットあるいはパケット誤り率を低減するために幅広く用いられている。多くの応用形態では、この目的を果たすために、たたみ込み符号が用いられる。たたみ込み符号は、ステートマシンによって定義され、状態の遷移は符号化されることになる入力ビットによって決定される。たたみ込み符号あるいはたたみ込み符号に基づく符号の場合、ビタビアルゴリズムと最大事後確率（ＭＡＰ）アルゴリズムの２つが、最もよく用いられる復号化アルゴリズムである。
【０００３】ビタビアルゴリズムはA. J. Viterbiによる「Error bounds for convolutional codes and an asymptotically optimum decoding algorithm」（IEEE Trans. Inform. Theory, Vol. IT-13, pp. 260-269, April 1967）に記載されており、参照して本明細書に援用している。そのアルゴリズムは、トレリスと呼ばれる時間的に展開される状態遷移図を通して最も可能性の高いパスを見つけることにより、受信されたビットあるいはシンボルを復号化する。
【０００４】図１は、従来のブロック直列ビタビ復号化器１００を示す。復号化器１００はＳ個の分岐メトリックユニット（ＢＭＵ）１０２を備えており、それぞれ１０２－ｊで示されている（ただしｊ＝０、１、．．．Ｓ－１である）。ＢＭＵ１０２－ｊのうちの所与の１つのものは、１ビットあるいはシンボルを受信する入力と、対応する加算－比較－選択ユニット（ＡＣＳＵ）１０４－ｊの入力に接続される出力とを備える。また、ＢＭＵ１０２－ｊおよびＡＣＳＵ１０４－ｊは、図面においてそれぞれＢＭＵ$_j$およびＡＣＳＵ$_j$で示される。ＡＣＳＵ１０４の組の出力は、受信されたビットあるいはシンボルから復号化されたビットを生成する残存メモリユニット（ＳＭＵ）１０６に加えられる。ビタビ復号化器１００の動作が以下に記載される。この説明は、復号化される所与のブロックがＮ個の受信されたビットあるいはシンボルを含むものと仮定している。
【０００５】復号化器１００は、状態が０のパスメトリックが０になり（$PM_0=0$）、それ以外の全ての状態のパスメトリックが無限大になるように（$PM_1,\ldots,s-1=\infty$）初期化することにより初期化される。その後、復号化アルゴリズムが、受信されたビットあるいはシンボル毎に繰り返す外部ループと、トレリス内の状態毎に繰り返す内部ループとを含む以下の再帰的な繰返しを実行する。
【０００６】ブロック内で、受信されたビットあるいはシンボルRx_i（ただしｉ＝０、１、．．．Ｎ－１）毎に、トレリス内の各状態に対して、
１．現在の状態から可能な次の状態までの各分岐に対する分岐メトリックを計算する。トレリスの所与の分岐のための分岐メトリックは、受信されたシンボルあるいはビットRx_iを与えるときに、現在の状態から、その分岐が接続する状態に遷移する可能性に関する指標である。分岐メトリック計算はＢＭＵ１０２によって実行される。
【０００７】２．各状態に入る最小パスメトリックを見つけるために比較を実行する。パスメトリックは、以前の状態パスメトリックと、その遷移に対して関連する分岐メトリックとの和として形成される。ここで、この最小値は、次の繰返しのための状態パスメトリックになる。比較は、ＡＣＳＵ１０４において実行される。
【０００８】３．各状態に対して、比較によってどの分岐に到達したかの判定をＳＭＵ１０６に格納する。
【０００９】上記のように、この繰返しプロセスは、復号化されることになるＮビットあるいはシンボルからなる受信されたブロックにおいてビットあるいはシンボル毎に１回実行され、各ビットあるいはシンボルに対して、受信された系列の１つの更新された推定値を生成する１回の復号化の繰返しを完了する。スループットを最大にするために、状態並列アーキテクチャを用いることができ、図１に示されるように、内部ループ動作が多数の状態に対して並行して実行される。しかしながら、外部ループ動作は並行には実行されない。なぜなら、１回の繰返しから得られる各状態のための更新されたパスメトリックが、次の繰返しのための入力として必要とされるからである。
【００１０】その復号化アルゴリズムは、受信されたブロックの全てのビットあるいはシンボルが処理された後に、最終的に最小のパスメトリックを有する状態を特定することにより終了する。この状態は、獲得状態（winning state）と呼ばれる。獲得状態から、そこで格納された復号化判定は、その状態を先行する状態にトレースバックされる。このトレースバックプロセスは、そのブ

ロックの開始時に対応する状態に到達するまで継続される。これらの判定によって定義されるパスは、ビットあるいはシンボルに関する最も伝送された可能性の高い系列を特定し、復号化器出力を生成するために処理される。

【００１１】ビタビアルゴリズムとは異なり、上記のＭＡＰアルゴリズムは、伝送された系列内の各ビットあるいはシンボルの最大事後確率を見つけるために、トレリスにわたる順方向および逆方向パスを用いる。ＭＡＰアルゴリズムは、L. R. Bahl, J. Cocke, F. JelinekおよびJ. Ravivによる「Optimal decoding of linear codes for minimizing symbol error rate」（IEEE Trans. Inform. Theory, Vol. IT-20, pp. 284-287, March 1974）に、より詳細に記載されており、参照して本明細書に援用している。

【００１２】ビタビおよびＭＡＰアルゴリズムはいずれも順次処理に基づいており、以前のビットあるいはシンボルに対するＡＣＳ計算の結果に応じて、受信されたブロック内の各ビットあるいはシンボルに対して動作が実行される。この依存性により，ビットあるいはシンボルレートより高いＡＣＳ動作のパイプライン化が妨げられ、それゆえ、たたみ込み符号を復号化することができる速度を制限し、それにより、復号化器のスループットを制限する。

【００１３】より高い誤り訂正能力を必要とする応用形態では、多くの場合に、２つ以上のたたみ込み符号が直列あるいは並列に連結して用いられる。連結される符号の一例が、たとえば、C. Berrou, A. GlavieuxおよびP. Thitimajshimaによる「NearShannon limit error-correcting coding: Turbo codes」（Proc. IEEE Int. Conf. Comm., Geneva Switzerland, 1993, pp. 1064-1070）に記載される、いわゆるターボ符号であり、参照して本明細書に援用している。連結された符号の復号化は、後続の符号トレリスのために、次の符号トレリスを次々に復号化するための入力として、１つの符号トレリスを復号化した結果を必要とする。そのような繰返し直列処理は、何度も、ビタビあるいはＭＡＰ復号化器のいずれかが各トレリスを通して何度も順次にトレースし、後続する繰返しは、以前の復号化の結果を待つ必要がある。ハードウエアで実装されるとき、復号化アルゴリズムの順次実施される性質は、ビットあるいはシンボル直列アーキテクチャを必要とする。これは、実行される繰返しの数および構成する符号の数だけビタビあるいはＭＡＰ復号化器の待ち時間を増やし、それにより、連結された符号を復号化するための待ち時間が相当に長くなる。

【００１４】上記のように、ビタビおよびＭＡＰアルゴリズムの再帰的な依存性によって、復号化器スループットを改善するために計算をパイプライン化することも不可能になる。スループットを改善し、待ち時間を最小にするために、より速いクロック速度で復号化器を駆動することはできるが、そのようなアプローチは復号化器の電力の浪費を増加させる。連結された符号を復号化することに関連する電力および待ち時間の問題は、多くの場合に、実用的な応用形態に対してそのアプローチを用いることを制限する。さらに、そのような符号が用いられる場合であっても、繰返しの数が制限され、それにより待ち時間に関する要件を満たすために符号化利得を犠牲にする場合がある。より少ない待ち時間で、そのような符号に対する復号化の各繰返しを実行できたなら、復号化器の性能は、実行される繰返しの数を増やすことにより改善することができるであろう。

【００１５】また、確率依存グラフ（probability dependency graph）を用いて、特定のクラスの符号、たとえば複合符号のブロック直列復号化プロセスを考案できることも当業者には知られている。複合符号は一般に、多数の符号の組み合わせから生成される符号のことである。たとえば、参照して本明細書に援用している、F. R. KschischangおよびB. J. Freyによる「Iterative decoding of compound codes by probability propagation in graphical models」（IEEE Journal on Selected Areas in Comm., Vol. 16, No. 2, pp. 219-230, Feb. 1998）を参照されたい。しかしながら、これまでそのような技術は、たたみ込み符号、ターボ符号あるいはその他の連結された符号等を含む、汎用の符号のブロック並列復号化に適用されてこなかった。ただし、汎用の符号は、パリティ検査行列を判定することができる任意の符号を含む。

【００１６】上記の説明から明らかなように、電力の浪費を増加させることなく、スループットを改善し、待ち時間を低減する改善された復号化アルゴリズムおよび対応する復号化器アーキテクチャが必要とされている。

【００１７】

【発明が解決しようとする課題】したがって、本発明の目的は、たとえば、たたみ込み符号、ターボ符号あるいはたたみ込み符号またはブロック符号等に基づくその他の連結された符号を含む、多種多様な符号の繰返し復号化を実行するためのブロック並列復号化アルゴリズムおよび対応する復号化器アーキテクチャを提供することである。

【００１８】

【課題を解決するための手段】本発明は、汎用の符号に対する復号化器の問題をブロック並列形式に定式化し直し、上記の従来のブロック直列復号化アルゴリズムおよび復号化器アーキテクチャに固有の直列依存性を排除し、それにより、実質的に待ち時間を低減し、スループットを高める点で有利である。

【００１９】本発明によれば、ブロック並列復号化アルゴリズムおよび対応する復号化器アーキテクチャは、確率依存グラフの形に構成された１組の相互接続された処

理ノードを用いる。確率依存グラフは、ビットあるいはシンボルからなるブロックを符号化するために用いられる符号によって少なくとも部分的に特徴づけられ、処理ノードが、復号化されることになるビットあるいはシンボルからなるブロックのためのブロック並列復号化プロセスを実施する。

【００２０】本発明の例示的な実施形態では、確率依存グラフは、Ｎ個一組の変数ノードとＴ個一組の検査ノードとを含む２つからなる確率依存グラフの形をなし、Ｎ個の変数ノードのうちの１つが復号化されることになる所与のブロックのＮ個のビットあるいはシンボルそれぞれに関連する。ブロック並列復号化プロセスの１回の繰返しは、変数ノード内に、所与のブロックにおける全ビットあるいはシンボルのための更新された推定値を生成する。この場合のブロック並列復号化プロセスは、変数ノードに接続される各検査ノードに関数ｆ（Ｒｘ$_i$）を送信することにより開始される。ただしＲｘ$_i$は、受信されたビットあるいはシンボルで、ｉ＝０、１、．．．Ｎ－１であり、ｆ（.）は、閾値関数、線形換算関数、あるいは近似指数関数のような、符号によって少なくとも部分的に決定される関数である。

【００２１】ブロック並列復号化プロセスは、検査ノードに接続される変数ノードからの入力の関数ｇ（.）を各検査ノードにおいて並行して計算することと、これらの検査ノードの計算結果を接続される変数ノードに送信することとを含む。関数ｇ（.）は、たとえば、パリティ検査関数、対数領域の双曲正接関数、あるいは対数領域の最大関数を含む場合がある。たとえば、関数ｇ（.）は、パリティ検査関数、さらに現在の入力がパリティ検査の制約条件を満たす信頼性の指示値を更新するための関数を含む場合がある。付加的な信頼性更新関数には、対数領域の双曲正接関数を用いることができる。

【００２２】ブロック並列復号化プロセスはさらに、各変数ノードにおいて、その変数ノードに接続される検査ノードからの入力の関数ｈ（.）を並行して計算することと、これらの変数ノードの計算結果を、接続される検査ノードに送信することとを含む。関数ｈ（.）には、たとえば、多数決関数あるいは平均化関数を用いることができる。

【００２３】ブロック並列復号化プロセスは、全ての検査ノードが符号の制約条件を満たしたとき、あるいは最大数だけ繰り返した後に終了する。終了時に、変数ノードの値が、推定、すなわち復号化される伝送ビットあるいはシンボルと、おそらく関連する事後確率あるいは信頼性とを決定する。

【００２４】本発明の別の例示的な実施形態では、確率依存グラフは、初期レベル、少なくとも１つの中間レベルおよび最終レベルを含む多数レベルのノードを有する有向確率依存グラフ（directional probability dependency graph）の形をとり、順次的なブロック内依存性を有することなく、各レベルにおいて計算を実行できるように構成される。初期レベルのノードは、復号化されることになる所与のブロック内の各ビットあるいはシンボルのための入力ノードを含む。各中間レベルのノードは、以前のレベルのノードからのみ入力を受信し、次に続くレベルのノードにのみ出力を供給する。最終レベルのノードの出力は、所与のブロックのための伝送されたビットあるいはシンボルの推定値を与える。ブロック並列復号化プロセスは、この実施形態では、初期レベル内の各ノードへの入力として、関数ｆ（Ｒｘ$_i$）を与えることにより初期化される。ただし、Ｒｘ$_i$は受信されたビットあるいはシンボルで、ｉ＝０、１、．．．Ｎ－１であり、ｆ（.）は、閾値関数、線形換算関数、あるいは近似指数関数のような、符号によって少なくとも部分的に決定される関数である。その後、各中間レベルのノードは、以前のレベルのノードから到来する値の関数ｇ（.）を計算し、次のレベルのノードにその計算結果を渡し、各ノードが、所与のブロック内の全ビットあるいはシンボルのための更新された推定値を生成できるようにする。

【００２５】本発明の別の態様によれば、相互接続される処理ノードの組は、１つあるいは複数のプログラム可能な計算ユニットを用いて実装することもできる。たとえば、異なる計算ユニットを用いて、各ノードを実装することができるか、あるいは多数のノードからなる所与の組を実装することができる。そのような計算ユニットは、種々の符号のための種々のブロック並列復号化プロセスを実施するのに適した、構成を変更可能なハードウエアあるいは他の装置を含む場合がある。

【００２６】

【発明の実施の形態】ここで、多数の例示的な復号化アルゴリズムおよび対応する復号化器アーキテクチャを用いて、本発明が例示されることになる。本発明の特定の実施形態は単なる例示にすぎず、本発明は、確率依存グラフによるブロック並列復号化に関連し、スループットを改善し、かつ待ち時間を低減することにより利益を得ることができる任意の復号化アルゴリズムおよび対応する復号化器アーキテクチャに、より幅広く適用することができることを理解されたい。

【００２７】本明細書で用いられる用語「汎用の符号」は、一例であり、限定はしないが、関連するパリティ検査行列が構成されるか、導出されるか、そうでなければ判定される場合がある任意の符号を含むことを意図している。

【００２８】用語「確率依存グラフ」は、一例であり、限定はしないが、所与のノードに関連する値の厳密な、あるいは近似的な確率が、その所与のノードに接続される１つあるいは複数のノードに関連する１つあるいは複数の値の厳密な、あるいは近似的な確率に関して表され

ることができるように構成された、相互接続されるノードの任意の表現を含むことを意図している。

【００２９】例示的な実施形態において、本発明は、確率依存グラフによって表される符号を復号化するためのブロック並列アルゴリズム、および対応するブロック並列復号化器アーキテクチャを提供する。本発明の一態様によれば、所与の復号化器アーキテクチャは、確率依存グラフを物理的に具体化したものを表しており、グラフ内のノードは計算ユニットあるいはその適切な部分に対応しており、グラフ内のエッジはノード間の接続に対応する。本発明は、確率依存グラフがブロック内に順次的な依存性を持たず、それゆえ、ブロック並列およびパイプライン化の両方の復号化アルゴリズムおよびアーキテクチャを開発するための原理として用いることができるという、本発明者の認識に一部基づいている。本発明による復号化器アーキテクチャは、図１とともに記載されたような従来のブロック直列復号化器アーキテクチャと比較すると、スループットと待ち時間に関して相当な利点を提供する点で有利である。

【００３０】図２および図７は、本発明によるブロック並列復号化アルゴリズムのための２つの例示的な復号化器アーキテクチャを示す。これらの例示的なアーキテクチャでは、復号化されるブロックは、Ｎ個の受信されたビットあるいはシンボルを含むものと仮定する。

【００３１】図２は、本発明による２ノード型のブロック並列確率依存グラフ復号化器２００の例示的な実施形態を示す。復号化器２００は、２つのタイプのノード、すなわち変数ノードおよび検査ノードを含む２ノード型の確率依存グラフの形で実装される素子を含む。Ｎ個の全変数ノードの場合に、所与のブロック内の全受信ビットあるいはシンボルが復号化されるために、１つの変数ノードが必要とされる。検査ノードの数はＴで示され、その数は符号、およびブロック内のビットあるいはシンボルの数によって決定される。変数ノードおよび検査ノードの接続性は、符号によって決定される。

【００３２】図２の復号化器２００では、変数ノードは白丸によって示され、検査ノードは黒丸によって示される。この実施形態における各ノードは、以下に記載される動作のうちの１つあるいは複数の動作を実行することができる計算ユニットに対応する処理ノードであると仮定される。しかしながら、変数ノードおよび検査ノードは、たとえば、スループット要件が変数ノードおよび検査ノードの処理速度よりも低い場合に、同じ処理ノードを用いて、多数のノードが評価されるようにグループ化されることを理解されたい。そのような構成では、１つの計算ユニットを用いて、確率依存グラフの多数のノードを実装することができる。したがって、所与の処理ノードは、１つの専用の計算ユニット、あるいはそのようなユニットの一部を用いて実装することができる。したがって、ここで記載される復号化器２００および他の復号化器内の各ノードは、１つの計算ユニット、あるいは多数ノード計算ユニットの一部を表す場合がある。

【００３３】復号化器２００は、変数ノードに接続される各検査ノードに、受信された値の関数、たとえばｆ（Ｒｘ$_i$）を送信することにより初期化される。ただし、Ｒｘ$_i$は受信されたビットあるいはシンボルで、ｉ＝０、１、．．．Ｎ－１であり、ｆ（.）は、符号によって決定される関数、たとえば閾値関数、線形換算関数あるいは近似指数関数である。この初期化は、図２の変数ノードの左側に示される。その後、復号化器２００は以下の動作を繰り返す。

【００３４】１．Ｃ$_0$、Ｃ$_1$、．．．Ｃ$_{T-1}$で示される各検査ノードでは、その検査ノードに接続される変数ノードからの入力のおそらく異なる関数ｇ（.）を計算する。たとえば、異なる検査ノードでは異なる関数ｇ（.）が用いられる場合があるか、あるいは用いられる関数は、繰返しの数あるいは別のパラメータを用いて変更することができる。所与の１つの検査ノードにおける関数ｇ（.）には、たとえば、パリティ検査関数、対数領域の双曲正接関数あるいは対数領域の最大関数を用いることができる。これらの計算の結果を接続される変数ノードに送信する。その結果は通常、接続される各変数ノードの場合に異なる値である。これらの計算は、並行して実行される場合もある。

【００３５】２．Ｖ$_0$、Ｖ$_1$、．．．Ｖ$_{N-1}$で示される各変数ノードでは、その変数ノードに接続される検査ノードからの入力のおそらく異なる関数ｈ（.）を計算する。たとえば、異なる変数ノードでは異なる関数ｈ（.）が用いられる場合があるか、あるいは用いられる関数は、繰返しの数あるいは別のパラメータを用いて変更することができる。所与の１つの変数ノードにおける関数ｈ（.）には、たとえば、多数決関数あるいは平均化関数を用いることができる。これらの計算の結果を接続される検査ノードに送信する。同様に、これらの計算は並行して実行される場合がある。

【００３６】このプロセスの１回の繰返しは、受信されたブロック内の全ビットあるいはシンボルに対して、１つの更新された推定値を生成する。所与の繰返しの各計算は、従来の復号化アルゴリズムおよびその対応するアーキテクチャの順次的なブロック内依存性を有することなく、ブロック内で並列に実行することができる点で有利である。

【００３７】復号化器２００の復号化アルゴリズムは、全ての検査ノードが符号の制約条件を満たしたときに、あるいは最大数だけ繰り返した後に終了され、変数ノードの計算結果は、その変数のための伝送された値の復号化された推定値を与える。繰返しの最大数は一般に特定の実装形態に依存し、たとえば、特定の実装形態において利用可能な時間内で、どれだけ多くの繰返し数を達成することができるかに対応する場合がある。

15

【００３８】復号化器２００の確率依存グラフは、対応する符号に関連するパリティ検査行列Ｈを用いて構成される場合があり、１つの変数ノードがＨの各列に関連し、１つの検査ノードがＨの各行に関連する。Ｈの全組の要素$h_{n\ m}$は、変数ノードｎと検査ノードｍとの間の接続に対応する。それゆえ、復号化器内の変数ノードｎに入るか、あるいは変数ノードｎを離れる、グラフ内のエッジの数は、Ｈの列ｎにおける０ではないエントリの数に等しく、復号化器内の検査ノードｍに入るか、あるいは検査ノードｍを離れる、グラフ内のエッジの数は、Ｈの行ｍにおける０ではないエントリの数に等しい。Ｈにおいて行演算を実行して、グラフ内のエッジの数を低減し、復号化器の構造あるいは性能を改善することもできる。

【００３９】変数ノードにおいてその時点で復号化されたビットx_i（ただし$i \in \{0, 1, \ldots N-1\}$が以下のパリティ検査行列を満足するか否かを試験することにより、復号化器２００において完了のための試験を実行することができる。

$H \cdot x = 0$,

ただし、ｘは復号化されたビットのベクトル$\{x_0, x_1, \ldots x_{N-1}\}$であり、０はｍビットゼロベクトルであり、ｍはパリティ検査行列の行の数と、検査ノードの数とに対応する。別の完了のための試験は、全ての検査ノードがその時点の入力によって満足されることを試験することにより実行することができる。

$chk_i = 0 \forall i \in \{0, 1, \ldots, K-1\}$,

ただし、chk_iは、検査ノードｉに対する全入力の排他的論理和である。

【００４０】図３および図４は、図２の復号化器２００の硬判定実装形態における、それぞれ変数ノードおよび検査ノードの例を示す。図３を参照すると、所与の変数ノード３００は、１組の要素３０２－０、３０２－１、３０２－２、．．．３０２－ｋを含み、それぞれ１組の入力の特定の部分の重み付けされた多数決関数を実施する。付加的な要素３０４は、１組の入力全体の重み付けされた多数決関数を実施する。重み付けされた多数決関数要素３０２－０、３０２－１、３０２－２、．．．３０２－ｋはそれぞれ、１組の２対１マルチプレクサ３０６－０、３０６－１、３０６－２、．．．３０６－ｋのうちの対応する１つのマルチプレクサに判定出力を供給する。マルチプレクサはそれぞれ共通の開始信号によって駆動され、その出力は、共通のクロック信号によって駆動されるＤフリップフロップ３０８－０、３０８－１、３０８－２、．．．３０８－ｋにラッチされる。多数決関数要素３０４の出力は、Ｄフリップフロップ３１０にラッチされ、その出力は判定信号ｄｅｃ。を表す。

【００４１】図４を参照すると、所与の検査ノード４００は、１組の排他的論理和（ＸＯＲ）ゲート４０２－０、４０２－１、４０２－２、．．．４０２－ｋを備え、それぞれ１組の入力in_0、in_1、in_2、．．．in_kのうちの１つを受信し、対応する１組の出力out_0、out_1、out_2、．．．out_kのうちの１つを生成する。検査ノード４００はさらに、入力としてin_0、in_1、in_2、．．．in_kをそれぞれ受信し、出力として上記のｃｈｋｉ信号を生成する付加的なＸＯＲゲート４０４も備える。その後、この出力は、各ＸＯＲゲート４０２－０、４０２－１、４０２－２、．．．４０２－ｋに入力として供給される。

【００４２】図５および図６は、図２の復号化器２００の軟判定実装形態における、変数ノードおよび検査ノードをそれぞれ示す。軟判定復号化器では、渡される各メッセージの符号に関連する信頼性が存在する。それゆえ、検査ノードは、硬判定復号化器に存在し、図４に示されるタイプのパリティ検査に加えて、信頼性を更新するように構成される回路も備える。

【００４３】図５を参照すると、軟判定復号化器のための変数ノード５００の更新部分は、加算器５０２、および１組の減算器５０４－０、５０４－１、．．．５０４－ｊを備える。各変数ノードV_i（ただし$i \in \{0, 1, \ldots N-1\}$内では、受信された値$Rx_i$は、関数ｆ（.）、たとえば線形換算関数によって演算され、結果Λ_iを生成する。その後、図５の変数ノード更新が、以下の式にしたがって実行される。

【数１】

$$out_0 = \frac{1}{k}\left(\Lambda_i + in_0 + in_1 + \ldots + in_j - in_0\right)$$

$$out_1 = \frac{1}{k}\left(\Lambda_i + in_0 + in_1 + \ldots + in_j - in_1\right)$$

$$\vdots$$

$$out_j = \frac{1}{k}\left(\Lambda_i + in_0 + in_1 + \ldots + in_j - in_j\right)$$

ただしｋは正規化定数である。図５の回路のこれらの出力は、上記の信頼性を表す。

【００４４】図６は、本発明による軟判定復号化器内の検査ノード６００の信頼性更新回路部分の一例を示す。検査ノード６００は、１組の入力関数要素６０２－０～６０２－ｋと、１組の出力関数要素６０４－０～６０４－ｋと、加算器６０６と、１組の減算器６０８－０～６０８－ｋとを備える。入力関数要素は、この例では双曲正接の自然対数を実施し、一方、出力関数要素は相補的な演算を実施する。加算器６０６は、入力関数要素それぞれからの出力を受信し、生成された和を各減算器に供給する。その和と個々の関数要素出力との差が、出力関数要素に適用され、対応する更新された出力が生成される。

【００４５】図６に示される実装形態では、たとえば、双曲正接とは対照的に、ｍａｘ－ｌｏｇ規則を用いて検査ノードを更新することにより簡略化することができ

る。この簡略形態を用いると、各出力ｉのための信頼性は、対数領域において実行される、入力ｉを除く、全ての入力の最大の信頼性として与えられる。

【数２】

$$out_i = \exp\left(\max_{j,j\neq i}\left(\ln(in_j)\right)\right).$$

【００４６】図７は、本発明によるブロック並列ベイズネットワーク確率依存復号化器７００の例示的な実施形態を示す。復号化器７００は、有向ベイズネットワーク確率依存グラフの形で実装される要素を含む。そのグラフは、復号化されることになるブロック内の全ビットあるいはシンボルのための入力ノードを含む。これらの入力ノードは、図面においてレベル０で示される。そのグラフはさらに、任意のノードが以前のレベルからの入力のみを有し、次に続くレベルへの出力のみを有するような、ノードのレベルを含む。この構成によって、計算は、順次的なブロック内依存性を有することなく、各レベルにおいて実行されるようになる。復号化器の段数、レベル当たりのノード、およびノード間に必要とされる接続は、符号によって決定される。ノードの最終レベルは、レベルＪによって示され、伝送された系列の推定値y_0、．．．、y_{K-1}を生成する。

【００４７】復号化器７００は、レベル０の各ノードへの入力として、受信された値の関数、たとえばｆ（Rx_i）を与えることにより初期化される。ただし、Rx_iは受信されたビットあるいはシンボルで、ｉ＝０、１、．．．Ｎ－１であり、ｆ（.）は、符号によって決定される関数、たとえば、閾値関数、線形換算関数あるいは近似指数関数である。この初期化入力は、図７のレベル０ノードの左側に示される。

【００４８】復号化器７００の各レベルは以前のレベルから到来する値のおそらく異なる関数ｇ（.）を計算し、この計算の結果は、復号化器７００の次のレベルに渡される。上記のように、関数ｇ（.）には、たとえば、パリティ検査関数、対数領域の双曲正接関数、あるいは対数領域の最大関数を用いることができる。

【００４９】この復号化プロセスの１つのレベルは、受信されたブロック内の全ビットあるいはシンボルに対して更新された推定値を生成する。上記のように、復号化器の最終レベル、すなわちレベルＪは、伝送された系列の推定値y_0、．．．y_{K-1}を生成する。出力Ｋの数は一般に入力の数よりも小さい、すなわちＫ＜Ｎであろう。

【００５０】復号化器７００の確率依存グラフ内の情報の流れは、図２の２ノード型のグラフ復号化器２００の場合のような双方向の流れでないことは明らかである。再帰的な依存性を持たない、この有向構造は、非常に高いスループットの復号化器を製造するためにパイプライン化するのに非常に適している。

【００５１】図８は、本発明による硬判定有向ネットワーク確率依存グラフ復号化器８００の一例を示す。復号化器８００は、図７に示されるタイプの復号化器の特定の硬判定実装形態を表す。復号化器８００は、相互接続されるノードの多数のレベルを含み、変数ノードは白丸で、検査ノードは黒丸で示される。ノードの第１のレベルは、レベル０で示され、それぞれ関数ｆ（Rx_i）を受信する１組の変数ノードを含む。ただしRx_iは受信されたビットあるいはシンボルである。ノードの次のレベルはレベル１で示され、１組の検査ノードを含む。その後のレベルは、変数ノードのレベルと検査ノードのレベルとの間で入れ替わる。ノードの最終レベルはレベルＪで示され、伝送された系列の推定値y_0、．．．y_{K-1}を生成する１組の変数ノードを含む。復号化器のレベル内では、変数ノードと検査ノードとの数が異なる場合があることに留意されたい。

【００５２】復号化器８００内の最終レベルではない変数ノードのレベルの各変数ノードには、図に示されるようにＤフリップフロップの形で実装される場合があるラッチが関連する。たとえば、復号化器８００のレベル０の変数ノードには、ラッチ８０２－０、８０２－１、８０２－２、．．．８０２－（Ｎ－１）が関連する。レベル１のような、復号化器８００の検査ノードレベルでは、以前のレベルの各変数ノードに対して１つのバッファが存在する。たとえば、復号化器８００のレベル１は、全Ｎ個のバッファ８０４－０、８０４－１、８０４－２、．．．８０４－（Ｎ－１）を備える。各バッファは、以前のレベルの変数ノードのラッチの出力に接続される入力と、次に続くレベルの変数ノードのラッチの入力に接続される出力とを有する。

【００５３】復号化器８００の種々のレベルの変数ノードと検査ノードとの間の接続は、対応する符号のパリティ検査行列によって定義される。レベルの数は、その復号化器が実行する繰返しの数を決定する。復号化器８００内の変数ノードと検査ノードは、それぞれ図３および図４の２ノード型の硬判定確率グラフ復号化器の変数ノードおよび検査ノードと同じように構成することができる。

【００５４】本発明の復号化器アーキテクチャの接続性および計算ユニットは、当分野においてよく知られている、構成を変更可能なハードウエア技術を用いて、プログラム可能に行われ、それにより、同じ復号化器ハードウエア上で多くの異なる符号を復号化できるようにする。

【００５５】本発明は、たとえば、たたみ込み符号、ターボ符号、ハミング符号、積符号、低密度パリティ検査（ＬＤＰＣ）符号、一般化ＬＤＰＣ（ＧＬＤ）符号、タナーグラフ（Tanner Graph）、線形あるいは巡回ブロック符号、連結ブロック符号等を含む幅広い符号に適用す

ることができる。

【００５６】ここに記載された典型的な復号化器は、本発明の動作を例示することを目的としており、それゆえ任意の特定の実施形態あるいは一群の実施形態に本発明を限定するものと解釈されるべきでないことを再び強調しておきたい。たとえば、復号化器要素の特定の構成および相互接続性は、符号のような、用途に固有の要因によって変更されるであろう。さらに、ここで記載された確率依存グラフ復号化器の多くの異なるハードウエア実装形態が実現可能である。以下の請求の範囲内にある、これらのおよび多くの他の代替形態は当業者には明らかであろう。

【００５７】

【発明の効果】上記のように、本発明によれば、たたみ込み符号、ターボ符号あるいはたたみ込み符号またはブロック符号等に基づくその他の連結された符号を含む、多種多様な符号の繰返し復号化を実行するためのブロック並列復号化アルゴリズムおよび対応する復号化器アーキテクチャを実現することができる。

【図面の簡単な説明】

【図１】従来のブロック直列ビタビ復号化器を示す図である。

【図２】本発明の第１の例示的な実施形態による２ノード型のブロック並列確率依存グラフを示す図である。

【図３】図２に示されるように実装される硬判定復号化器において用いるための変数ノードの一例を示す図である。

【図４】図２に示されるように実装される硬判定復号化器において用いるための検査ノードの一例を示す図である。

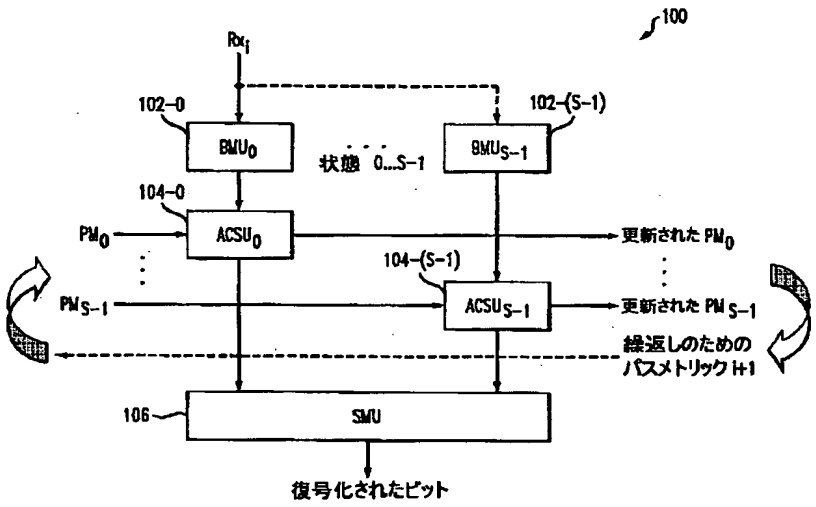
【図５】図２に示されるように実装される軟判定復号化器において用いるための変数ノードの一例を示す図である。

【図６】図２に示されるように実装される軟判定復号化器において用いるための検査ノードの一例を示す図である。

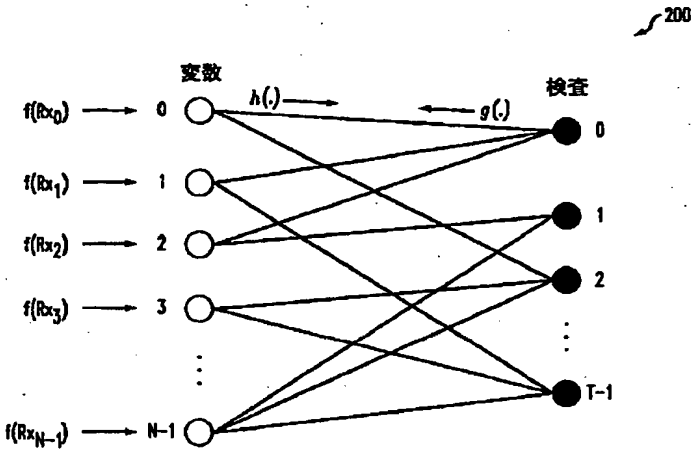
【図７】本発明の第２の例示的な実施形態によるブロック並列有向ネットワーク確率依存グラフ復号化器を示す図である。

【図８】図７に示されるタイプのブロック並列有向ネットワーク確率依存グラフ復号化器の例示的な硬判定実装形態を示す図である。

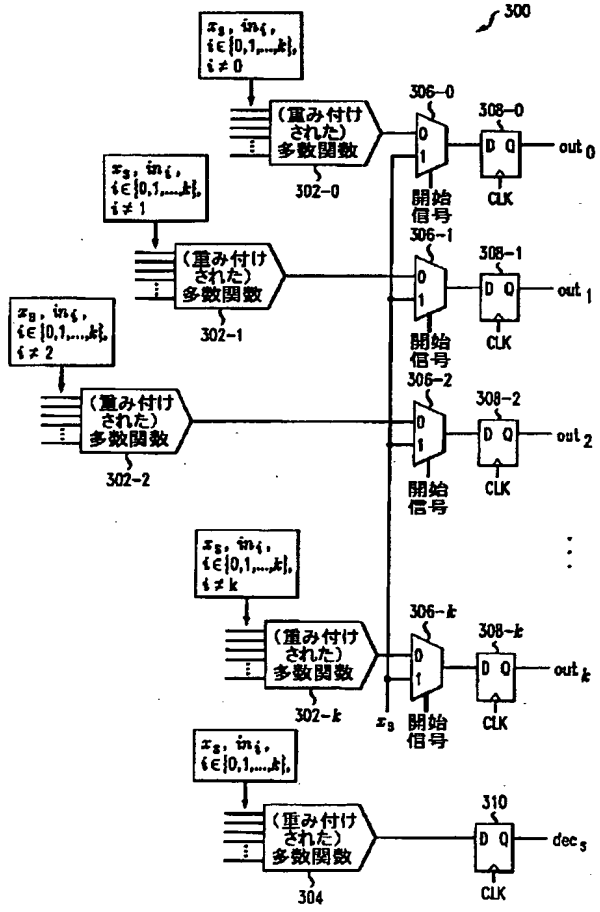
【図１】



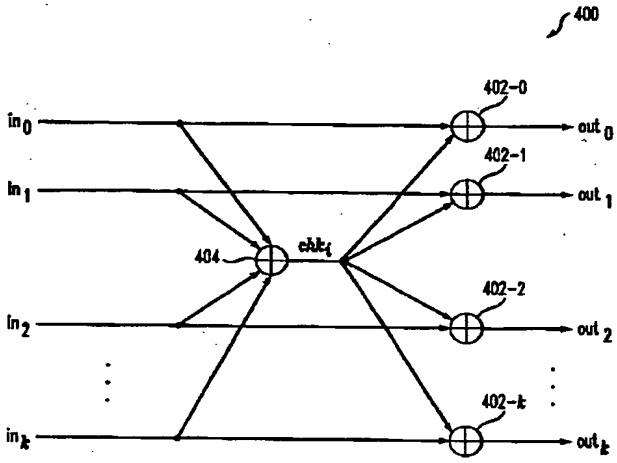
【図２】


200
変数
検査
h(.)
g(.)
f(Rx0)
f(Rx1)
f(Rx2)
f(Rx3)
f(RxN-1)
0
1
2
3
N-1
0
1
2
T-1

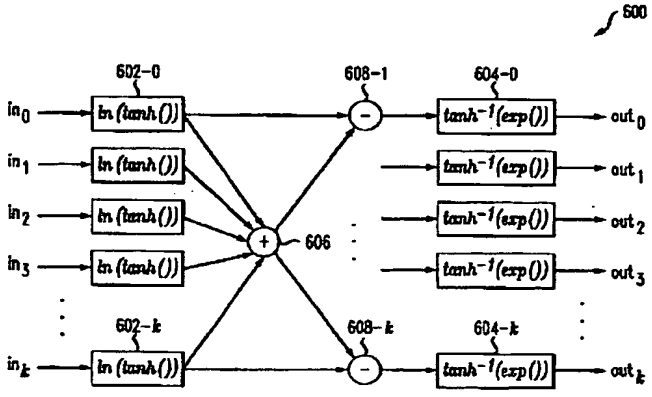
【図３】


300
(重み付けされた)多数関数
302-0
302-1
302-2
302-k
304
306-0
306-1
306-2
306-k
308-0
308-1
308-2
308-k
310
開始信号
CLK
out0
out1
out2
outk
decs

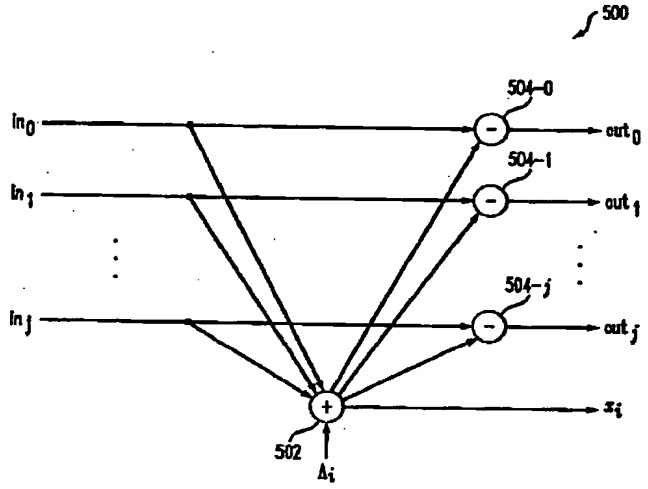
【図４】


400
402-0
402-1
402-2
402-k
404
in0
in1
in2
ink
out0
out1
out2
outk

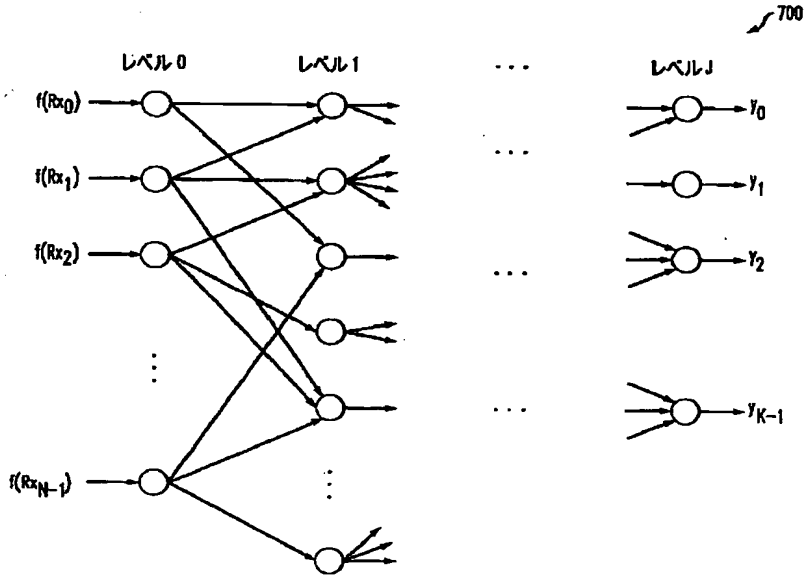
【図６】


600
602-0
602-k
604-0
604-k
606
608-1
608-k
ln(tanh())
tanh-1(exp())
in0
in1
in2
in3
ink
out0
out1
out2
out3
outk

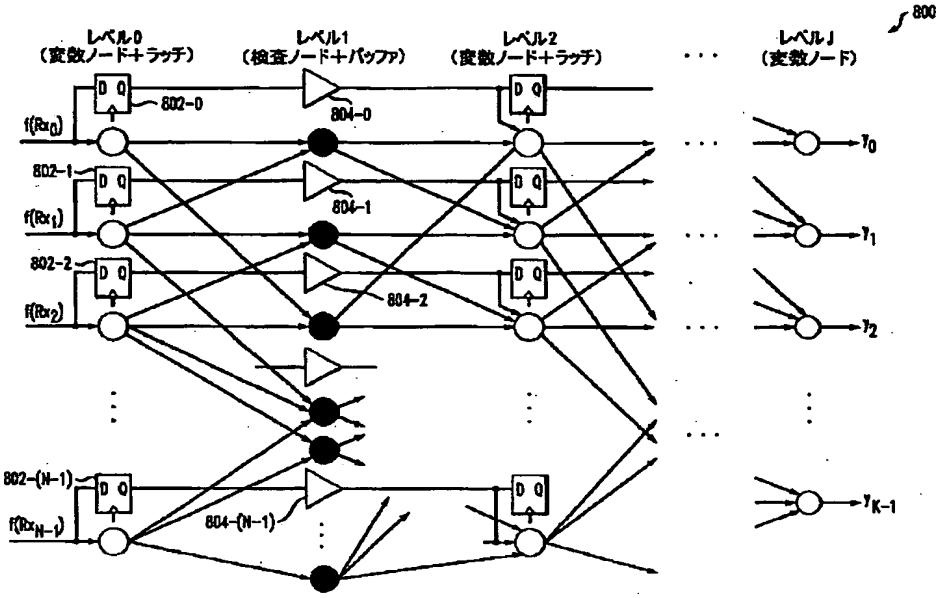
【図５】


500
504-0
504-1
504-j
in0
in1
inj
out0
out1
outj
xi
502
Ai

【図７】


700
レベル０
レベル１
レベルＪ
f(Rx0)
f(Rx1)
f(Rx2)
f(RxN-1)
y0
y1
y2
yK-1

【図８】



フロントページの続き

(72)発明者　アンドリュー　ジェー．ブランクスビー
アメリカ合衆国　07720　ニュージャーシィ，ブラッドレイ　ビーチ，オーシャンパーク　アヴェニュー　100　ピー

(72)発明者　クリストファー　ジョン　ハウランド
アメリカ合衆国　07266　ニュージャーシィ，マナラパン，エリオット　ロード　5

Ｆターム(参考)　5B001 AA01 AA10 AA13 AC01 AC04 AE02
5D045 DA20
5J065 AC01 AD01 AD10 AG05 AH01 AH03 AH07 AH11 AH12 AH15